Comparing Accuracy of Feature Selection Methods and Performance of Different Machine Learning Algorithms on Predicting Results on Breast Cancer Malignancy

Klaus Mana

University of Arizona

Abstract

Predicting the malignancy of Breast Cancer given features of the discovered tumor that we already know can be a useful tool for diagnosis. In this project, I have explored the application of multiple Machine Learning algorithms as it pertains to the problem of predicting whether a tumor is malignant or benign. I have also compared different feature selection methods, observing how they affect the accuracy of different models. Lastly, I have shown how Data Augmentation can lead to improvement in performance of a k-Nearest Neighbors Model. The Machine Learning algorithms I have used to apply to the data are k-Nearest Neighbors, Logistic Regression, Support Vector Classifier, and Random Forest.

1. Introduction

Predicting malignancy of tumor masses given their size, appearance, and other physical properties is a common application of machine learning (ML) in healthcare. While there are many other studies about refining the use of ML models and algorithms for this particular problem, I am using the plentiful available data on the matter [1] to explore more about ML models and how they compare to each other. I have tuned Random Forest (RF), k-Nearest Neighbors (kNN), Logistic Regression (LR), and Support Vector Classifier (SVC) models to see which one performs better given the Wisconsin Breast Cancer Dataset [1]. More importantly, I have chosen to explore how two different Feature Selection methods, Feature Selection by Random Forest as well as ANOVA affect the accuracy of these models, as well as how introducing more data points can help performance of models such as kNN. Feature Selection can be useful not just for increasing accuracy, but also significantly reducing performance costs on certain ML Models. Introducing more Data can be really useful in increasing the accuracy of a model if not enough data is available to begin with.

2. Background

The Data

The dataset [1] that I am using consists of 569 instances describing tumors, each of which contain a diagnosis and 30 real-valued features for each tumor, ranging from things like radius to fractal dimension [1]. The scikitlearn library [2] was used to obtain all of the models used.

The Models

RF is a model that constructs multiple decision trees during training and outputs the mode of the classes for classification tasks (such as our own, since we are classifying tumors as benign (B) or malign (M)). kNN classifies data based on the majority class of their k nearest neighbors in feature space. Logistic Regression is a linear model making predictions based on the probability of an instance belonging to a class based on a threshold. This one is the simplest of our models to train. The SVC model classifies data by finding the optimal hyperplane for separation.

Hyperparameters

Hyperparameters are the parameters that these models take which are provided rather than learned or calculated. A big part of this project was fine tuning these hyperparameters to increase model accuracy.

3. Methodology

I began this project by doing some exploratory work on the dataset itself, and trying to learn more about the features. During this process, I constructed different graphs such as boxplots or heatmaps (Figure 1).

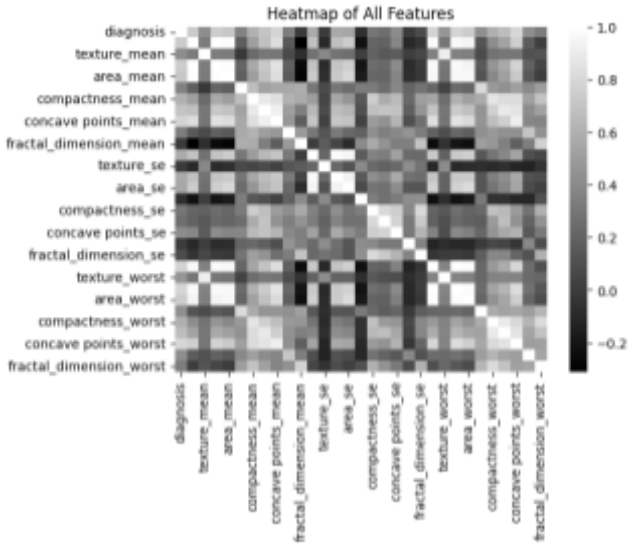


Figure 1

The depicted heatmap portrays the correlation between all of the different features. The lighter the color, the more correlated two features are to one another. This is helpful information to have since features strongly correlated to each other can prove to be redundant, and lead to increased complexity and performance costs for no real gain in model accuracy. Looking at this heatmap, I observed that there were quite a few features that seemed to be strongly correlated to other features. Therefore, one hypothesis that I explored was that by doing some careful feature selection, the performance cost of training models on this dataset could be reduced, while having no impact or minimal impact on the model accuracy. With this in mind, I decided to try two different classical methods for feature selection, RF driven feature selection and ANOVA/F-Test driven feature selection. To begin, I tuned a RF model with different values for the *number of estimators*

and *max features* hyperparameters. After doing this, I used sklearn's SelectFromModel [2] to rank features with my RF model and create a new feature list of only *9* features (instead of the original *30*), which were then used during training for other models. The choice to use this smaller list of features was to reduce performance cost. After RF, I tuned a kNN model. This was done by observing the performance against a validation set for an arbitrary range of k-values 1-60. Afterwards, given the results of this run, a smaller more feasible range of k values was selected, and the best k-value was chosen by using Cross-Validation on the training set. For the next two models, SVC and LR, GirdSearchCV [2] Cross-Validation was used to tune their hyperparameters.

ANOVA Feature Selection

I chose another feature selection method, ANOVA, to see what different features it would select and how they were different from the ones selected by the RF-driven feature selection. This time, I fine tuned the number of features that would be selected using a LR model. The ANOVA approach led to *19* features being selected, rather than the original list of *30*. All models were also tested against this feature list.

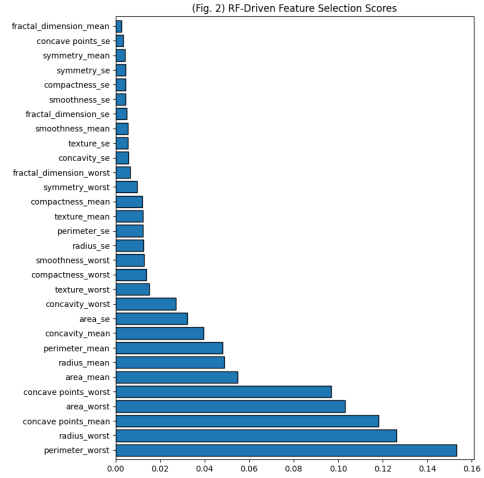
Data Augmentation

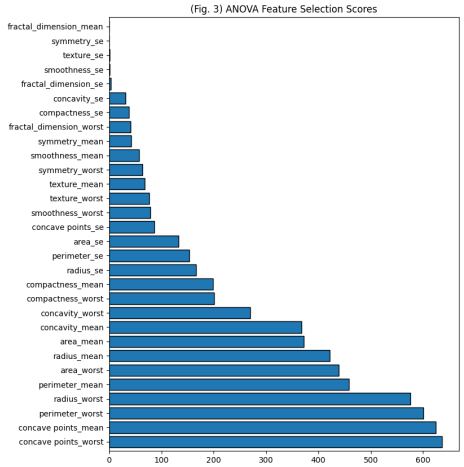
One more interesting hypothesis I wanted to test was how introducing more data may affect the kNN model. For this, I augmented the original dataset in three different ways: by adding data points with random noise introduced, by adding data points that were shifted, and by adding data points that were randomly scaled. I used the same hyperparameter for this kNN model as well, since the expected result of the Data Augmentation was just to introduce more data, similar to the data already available, so that the model can utilize it to better learn the problem.

4. Results

Feature Selection

Both the RF-driven feature selection and ANOVA came up with similar rankings of features based on importance. These rankings can be seen below in Figure 2 and Figure 3.

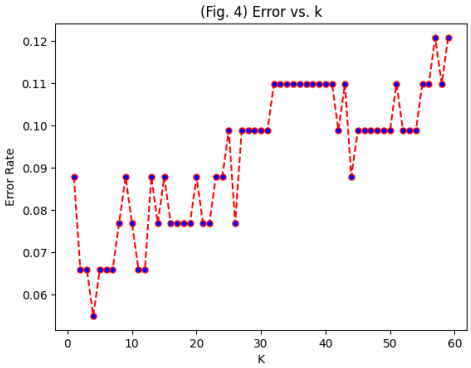




There was some contention between the two feature selection methods, mainly in the middle of the rankings. This proved to be relevant, as when tuning for the number of features, **19** features were picked by the ANOVA method, so some of these features that the RF-driven method deemed not useful were actually selected.

For the RF model, hyperparameter tuning did not prove all that useful, so **n_estimators=400** and **max_features=sqrt** were chosen more or less randomly.

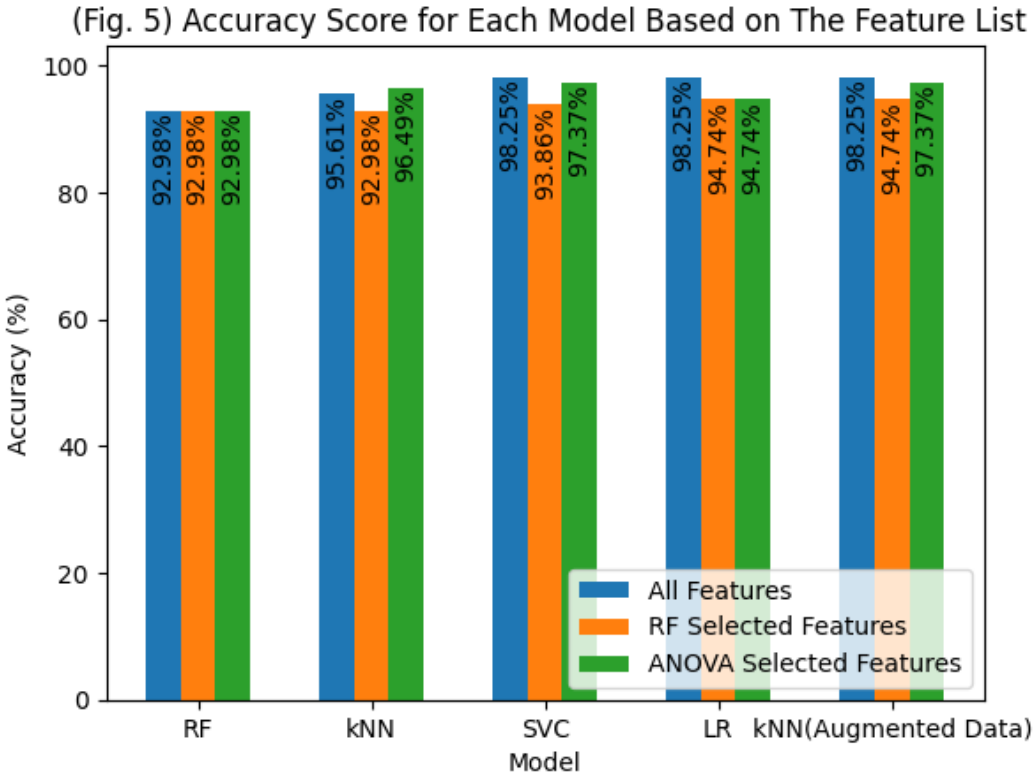
For kNN, the first round of tuning provided the Error Rates seen in Figure 4.



As can be seen, after approximately k=31, the error rate only increases. Therefore, Cross-Validation was done in the range 4-31, which led to deciding **k=8** as the most successful value for the kNN hyperparameter.

SVC and LR were both only tuned using GridSearchCV [2]. For SVC, **gamma=1e-4** and **C=1000** were picked. For LR, **C=1000** and **solver=liblinear** were chosen.

After tuning all the models, 3 different versions were fit for each: one using all of the available features, one using the ANOVA feature list, and one using the RF-driven feature list. All of these models were evaluated on the test set. Figure 5 shows the results of all evaluations.



As can be seen, using the full feature list proved to be, overall, the most successful, with one exception of kNN where the ANOVA features outperformed it. On this particular problem of Cancer Diagnosis, higher accuracy may be more preferred than lesser performance cost. However, the difference in accuracy is still very close in most cases, so it could be argued that the performance boost of using less than a third of the original list of features may be worth the decrease in accuracy.

The following table showcases the differences in accuracy between a benchmark notebook [3] and my models, tuned on the truncated feature list

Model	**Models Tuned on RF-driven feature list**	**Benchmark Notebook [3]**
RF	92.98%	**96%**
kNN	**98.25%** (after Data Augmentation)	95%
SVC	**98.25%**	97%
LR	**98.25%**	97%

As can be seen, tuning the hyperparameters and using a truncated feature list proved to be very useful for 3 out of 4 of the models that were trained.

The second hypothesis that I tried to explore was the effect of Data Augmentation on the kNN model. This proved to be very useful, as it substantially improved the accuracy of the kNN model, as can be seen in Figure 5. Another observation that can be made is how the kNN model with data augmentation matched the accuracy of the SVC and LR models (which had the highest accuracy to begin with), hinting to the idea that having more data points, even if generated by a rather naive approach such as the one used in this project, can prove very useful in improving model accuracy.

All of the code used for this project [4] can be found in the link in the reference list.

5. References

[1] Momeni, M. (2023, October), Breast Cancer Wisconsin (Diagnostic) Data Set. Retrieved Nov 3rd, 2023 from https://www.kaggle.com/datasets/imtkaggleteam/breast-cancer/data

[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.

[3] O_Bayram (2023, November), * Breast-Cancer-Prediction-0.991%. Retrieved Dec 8th, 2023 from https://www.kaggle.com/code/obayram28/breast-cancer-prediction-0-991

[4] Mana, K. (2023), GitHub repository, https://github.com/klaus-mana/machine-learning-final-project